AMENDMENTS TO THE CLAIMS

Amended claims follow:

1. (Currently Amended) A packet transmit queue control system, comprising:

a first data structure embodied on a tangible computer readable medium coupled to a packet controller and configured to store a plurality of first type packet pointers;

a second data structure embodied on a tangible computer readable medium coupled to the packet controller and configured to store a plurality of second type packet pointers, wherein the packet controller is configured to receive a first sequence of packet pointers and to provide each packet pointer to one of the first and second data structures; and

a port transmit controller coupled to the first and second data structures and configured to provide a second sequence of packet pointers;

wherein the first data structure includes a plurality of linked-list data structures.

2. (Cancelled)

3. (Original) The packet transmit queue control system of claim 1, wherein:

the second data structure includes a plurality of first-in first-out (FIFO) structures.

4. (Original) The packet transmit queue control system of claim 3, wherein:

each of the plurality of FIFO structures is coupled to a port.

5. (Original) The packet transmit queue control system of claim 1, wherein:

the first type packet pointers include unicast pointers; and

the second type packet pointers include multicast pointers.

6. (Original) The packet transmit queue control system of claim 1, wherein:

the second sequence of packet pointers includes port transmit scheduling information.

7. (Original) The packet transmit queue control system of claim 1, wherein:

an ordering of the second sequence of packet pointers is substantially consistent with a packet arrival order.

8. (Original) The packet transmit queue control system of claim 4, wherein:

the packet controller is configured to provide:

each of the plurality of first type packet pointers to a selected one of the plurality of linked-list data structures; and

each of the plurality of second type packet pointers to each or a group of the plurality of FIFO structures.

9. (Currently Amended) A data arrangement for packet transmit queue control, comprising:

a first data structure embodied on a tangible computer readable medium configured to store a plurality of first type packet pointers;

a second data structure embodied on a tangible computer readable medium configured to store a plurality of second type packet pointers; and

a third data structure embodied on a tangible computer readable medium coupled to the second data structure and configured to store a plurality of status flags;

wherein the first data structure includes a linked-list structure.

10. (Cancelled)

11. (Original) The data arrangement for packet transmit queue control of claim 9, wherein:

the second data structure includes a first-in first-out (FIFO) structure.

12. (Original) The data arrangement for packet transmit queue control of claim 9, wherein:

the first type packet pointers include unicast pointers; and

the second type packet pointers include multicast pointers.

13. (Original) The data arrangement for packet transmit queue control of claim 12, wherein:

each entry of the second data structure includes:

a previous unicast pointer indication field;

a next unicast pointer indication field;

a previous unicast pointer field; and

a packet pointer field.

14. (Original) The data arrangement for packet transmit queue control of claim 9, wherein:

the plurality of status flags includes:

a first type packet pointer head position indication;

a first type packet pointer tail position indication;

an overall head pointer indication; and

an overall tail pointer indication.

15. (Currently Amended) A method of inserting a packet pointer in a packet queue control system, comprising:

(a) determining if a pointer is a first type or a second type;

(b) if the pointer is the first type, determining if an overall tail is the first type or the second type;

if the overall tail is the second type, setting an overall tail flag to a first state, and setting an entry to the first state;

getting a first type tail;

linking the pointer in a first data structure to the first type tail; and

setting the first type tail to the pointer; and

(c) if the pointer is the second type, determining if the overall tail is the first type or the second type;

if the overall tail is the second type, adding the pointer to a second data structure field with a second state;

if the overall tail is the first type, adding the pointer to the second data structure field with the first state; and

setting the overall tail flag to the second state;

wherein the first data structure includes a plurality of linked-list data structures.

16. (Cancelled)

17. (Original) The method of inserting the packet pointer in the packet queue control system of claim 15, wherein:

the second data structure includes a first-in first-out (FIFO) data structure.

18. (Original) The method of inserting the packet pointer in the packet queue control system of claim 15, wherein:

the first type pointer includes a unicast packet pointer; and

the second type pointer includes a multicast packet pointer.

19. (Original) The method of inserting the packet pointer in the packet queue control system of claim 15, wherein:

the first state includes a yes-state; and

the second state includes a no-state.

20. (Original) The method of inserting the packet pointer in the packet queue control system of claim 15, wherein:

the first data structure is accessed at most once; and

the second data structure is accessed at most once.

21. (Currently Amended) A means for inserting a packet pointer in a packet queue control system, comprising:

(a) means for determining if a pointer is a first type or a second type;

(b) if the pointer is the first type, means for determining if an overall tail is the first type or the second type;

if the overall tail is the second type, a means for setting an overall tail flag to a first state, and a means for setting an entry to the first state;

a means for getting a first type tail;

a means for linking the pointer in a first data structure embodied on a tangible computer readable medium to the first type tail; and

a means for setting the first type tail to the pointer; and

(c) if the pointer is the second type, a means for determining if the overall tail is the first type or the second type;

if the overall tail is the second type, a means for adding the pointer to a field of a second data structure embodied on a tangible computer readable medium[[field]] with a second state;

if the overall tail is the first type, a means for adding the pointer to [[the]]a field of the second data structure [[field]]with the first state; and

a means for setting the overall tail flag to the second state;

wherein the first data structure includes a plurality of linked-list data structures.

22. (Currently Amended) A method of scheduling a packet pointer, comprising:

(a) determining if an overall head is a first type or a second type;

(b) if the overall head is the first type:

getting a first type head;

getting a first type pointer from a second data structure;

determining if the first type head matches the first type pointer;

if a match, setting an overall head flag to a second state; and

updating the first type head with a next pointer from a first data structure; and

(c) if the overall head is the second type:

getting a second type pointer from the second data structure;

determining if a field in the second data structure is a first state or the second state;

if the first state, setting the overall head flag to the first state; and

removing a head entry from the second data structure;

wherein the first data structure includes a plurality of linked-list data structures.

23. (Cancelled)

24. (Original) The method of scheduling the packet pointer of claim 22, wherein:
the second data structure includes a first-in first-out (FIFO) data structure.

25. (Original) The method of scheduling the packet pointer of claim 22, wherein:
the first type pointer includes a unicast packet pointer; and
the second type pointer includes a multicast packet pointer.

26. (Original) The method of scheduling the packet pointer of claim 22, wherein:
the first state includes a yes-state; and
the second state includes a no-state.

27. (Original) The method of scheduling the packet pointer of claim 22, wherein:
the first data structure is accessed at most once; and
the second data structure is accessed at most once.

28. (Currently Amended) A means for scheduling a packet pointer, comprising:
(a) a means for determining if an overall head is a first type or a second type;
(b) if the overall head is the first type:
a means for getting a first type head;
a means for getting a first type pointer from a second data structure;
a means for determining if the first type head matches the first type pointer;
if a match, a means for setting an overall head flag to a second state; and
a means for updating the first type head with a next pointer from a first data structure; and
(c) if the overall head is the second type:
a means for getting a second type pointer from the second data structure;
a means for determining if a field in the second data structure is a first state or the second state;

if the first state, a means for setting the overall head flag to the first state; and

a means for removing a head entry from the second data structure;

wherein the first data structure includes a plurality of linked-list data structures.

29. (Currently Amended) A method of managing packet pointers, comprising:

inserting a packet pointer according to the method of claim 15; and

scheduling the packet pointer comprising:

(a) determining if an overall head is a first type or a second type;

(b) if the overall head is the first type:

getting a first type head;

getting a first type pointer from a second data structure;

determining if the first type head matches the first type pointer;

if a match, setting an overall head flag to a second state; and

updating the first type head with a next pointer from [[a]]the first data structure; and

(c) if the overall head is the second type:

getting a second type pointer from the second data structure;

determining if a field in the second data structure is a first state or the second state;

if the first state, setting the overall head flag to the first state; and

removing a head entry from the second data structure.

30. (Currently Amended) The method of managing packet pointers of claim 29, wherein:

the inserting the packet pointer and the scheduling the packet pointer include using a data arrangement for packet transmit queue control, comprising:

[[a]]the first data structure configured to store a plurality of first type packet pointers;

[[a]]the second data structure configured to store a plurality of second type packet pointers; and

a third data structure coupled to the second data structure and configured to store a plurality of status flags.

31. (Currently Amended) The method of managing packet pointers of claim 29, wherein:

the inserting the packet pointer and the scheduling the packet pointer are operable in a system, comprising:

[[a]]the first data structure coupled to a packet controller and configured to store a plurality of first type packet pointers;

[[a]]the second data structure coupled to the packet controller and configured to store a plurality of second type packet pointers, wherein the packet controller is configured to receive a first sequence of packet pointers and to provide each packet pointer to one of the first and second data structures; and

a port transmit controller coupled to the first and second data structures and configured to provide a second sequence of packet pointers.

32. (New) The packet transmit queue control system of claim 1, wherein:
the port transmit controller is coupled to an output port.

33. (New) The packet transmit queue control system of claim 4, wherein:
each of the plurality of FIFO structures are coupled with a corresponding set of output port status flags.